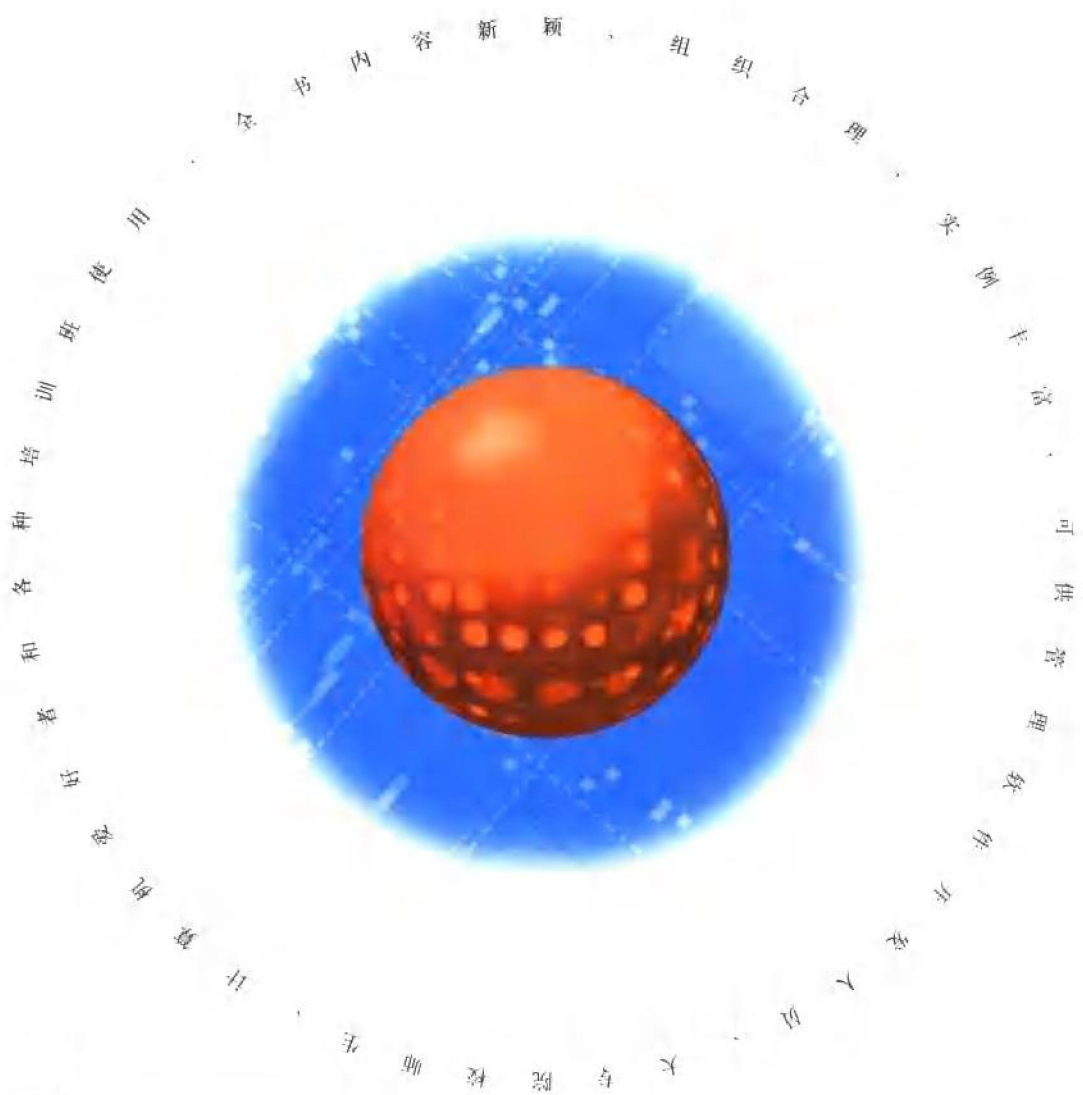
现代数据库技术丛书

中文 Visual FoxPro 5.0 开发指南

郭玲文 赵健雅 等编著



机械工业出版社

CMP

现代数据库技术丛书

中文 Visual FoxPro 5.0 开发指南

郭玲文　赵健雅　等编著

机械工业出版社

本书详细介绍了 Visual FoxPro 应用程序的开发过程，其内容包括 Visual FoxPro 程序设计的过程和方法、数据处理、界面设计、系统集成、客户/服务器程序开发、创建帮助文件等。

全书内容详尽、条理清晰，可供管理软件开发人员、大专院校师生、计算机爱好者和各种培训班使用。

图书在版编目（CIP）数据

中文 Visual FoxPro 5.0 开发指南/郭玲文，赵健雅主编 .-北京：机械工业出版社，1998.4

（现代数据库技术丛书）

ISBN 7-111-06170-5

Ⅰ.中… Ⅱ.①郭… ②赵… Ⅲ.关系数据库-数据库管理系统，FoxPro 5.0 Ⅳ.TP311.13

中国版本图书馆 CIP 数据核字（98）第 08221 号

出 版 人：马九荣（北京市百万庄大街 22 号 邮政编码 100037）

责任编辑：蒋 克

昌平环球印刷厂印刷 · 新华书店北京发行所发行

1998 年 4 月第 1 版第 1 次印刷

787mm×1092mm 1/16 · 24 印张

印数：0001－7000 册

定价：38.00 元

前　言

Visual FoxPro 5.0 是美国 Microsoft 公司推出的最新 32 位数据库管理系统，与 Visual FoxPro 以前版本相比，其特点如下：

- 增强的项目及数据库管理：在该版本中，用户可以对项目及数据进行更强的控制。例如，用户能够使用源代码管理产品（如 Microsoft Visual SourceSafe），同时在“项目管理器”中看到组件的状态。数据库容器允许几个用户在同一个数据库中同时创建或修改对象。
- 已改善的调试工具：在 Visual FoxPro 的这个版本中，用户可以更简便地调试及监控自己的应用程序组件。
- 更简便的表设计以及扩展的数据字典：在这个版本的“表设计器”中，用户可以在创建字段时方便地添加索引，也可以指定更多种默认值，它们可以使表的设计迅速而简便。
- 增强的查询及视图设计：现在用户可以创建外部联接，为列指定别名，选择最上面几条或百分之几的记录，所有这些功能都在“查询设计器”及“视图设计器”中。
- 增加的表单功能以及更简便的设计：数据字典的增强有助于表单的设计，“表单设计器”本身也更易使用并且提供了更多的功能。“表单设计器”支持单文档界面（SDI）以及多文档界面（MDI）的选项，因此用户的应用程序可以具有最希望的功能。
- 更多的向导：两个新的向导可以帮用户创建应用程序，“应用程序向导”可为用户的项目创建一个框架；“Oracle 升迁向导”可以把数据库、表及视图移到一个 Oracle 服务器上。
- OLE 与 ActiveX 更强的集成：Visual FoxPro 现在是一个 OLE 服务器，因此其他应用程序也可以利用 Visual FoxPro。ISimpleFrame 能力扩展了对更宽系列 ActiveX 控件的支持。Visual FoxPro 也提供创建自己的 OLE 服务器的能力，用户可以把这些服务器布置在本地或远程。
- 应用程序组件的实例：新的示例中收集了一系列应用程序组件，他们显示了如何利用 Visual FoxPro 的特性解决现实世界的问题。用户可以在应用程序中直接使用示例组件及它们的代码。

为了使用户能更好地使用 Visual FoxPro 5.0，我们特编撰了如下图书：《中文 Visual FoxPro 5.0 基础教程》介绍了 Visual FoxPro 5.0 的基本用法，内容包括对项目管理器、菜单设计器、表单设计器、数据库设计器等的介绍；《中文 Visual FoxPro 5.0 开发指南》则向读者集中介绍了一些与程序设计相关的高级主题，如 Visual FoxPro 程序设计方法、数据库规划、程序界面设计、共享程序设计、客户/服务器程序设计等；《中文 Visual FoxPro 5.0 属性、事件和方法参考手册》集中介绍了 Visual FoxPro 5.0 的类、对象、属性、事件、方法、系统变量和 API 函数等；《中文 Visual FoxPro 5.0 命令和函数参考手册》详细介绍了 Visual FoxPro 5.0 命令和函数等。

本书分别由郭玲文和赵健雅主编，此外，参与编写工作的还有刘畅、张阳、李腊梅、方志刚、何钦、林一明、王晨、严琳、钟雪萍、伊刚、李勇、杨昊、王敏、甘爽、蔼伟强、陈芸、肖晓、赵军、钱莹、刘岗、林晓东、袁鸣等。

编者

1998 年 1 月

目　录

第 1 章　Visual FoxPro 程序设计入门

Visual FoxPro 是一个功能强大的数据管理工具，您不仅可以通过交互方式利用它进行数据管理工作，而且可以创建应用程序来充分发挥它的全部功能。掌握 Visual FoxPro 的面向对象程序设计技术以及事件驱动模型，可以最大限度地提高程序设计的效率。

1.1　开发管理信息系统的特点

大家知道，FoxPro 是一种关系型数据库语言，其目的是用于开发管理信息系统(MIS)。自然，Visual FoxPro 也不例外，它只不过是对 FoxPro 以前版本进行了改进和功能扩充，使用户开发起来更容易。因此，在我们具体讲解 Visual FoxPro 程序设计之前，先花点时间了解一下管理信息系统的开发特点。

1.1.1　管理信息系统的开发过程

在开发 MIS 之初，首先面临的问题是如何划分开发阶段，并确定每个阶段的开发任务。这个问题的实质是：选择生存周期法进行开发，还是选择原型法进行开发。

1. 生存周期法

• 生存周期法的特点

生存周期法的特点是：开发人员在软件生存周期的每一阶段施行严格的定义。试图在每一阶段结束后，通过严格的阶段性审查/确认，得到该阶段一致、完整、正确、无二义性的良好文档资料，以此作为本阶段的结束标志和下一开发阶段的依据，从而形成一个理想的线性开发序列。以每一步的正确性和完整性来把错误消灭在萌芽阶段，从而减少系统的修改量，保证最终产品质量。

• 生存周期法的开发阶段划分

选择生存周期法开发 MIS，可将开发过程划分为六个阶段。

① 软件计划。在计划阶段，确定要开发软件的总目标，给出它的功能、性能、可靠性以及接口等方面的设想。研究完成该项软件任务的可行性分析，探讨出解决问题的方案。并且对可供使用的资源、成本、可取得的效益和开发的进度作出估计，以及制定完成开发任务的实施计划。

② 需求分析。在需求分析阶段，对开发的软件进行详细的定义。这应由软件开发人员和用户共同讨论决定。要确定哪些需求是可以满足的，哪些需求难以满足，并分别加以确切的描述。在这个阶段，还要写出软件需求说明书以及初步的系统用户手册，提交管理机构评审。

③ 软件设计。软件设计是 MIS 工程的技术核心。在这个阶段，设计人员要把已确定了的各项需要转换成一个相应的体系结构，结构中每一组成部分是意义明确的模块，每个模块都和某些需求相对应，这就是概要设计。对每个模块要完成的工作进行具体的描述，为程序编写打下基础，这就是详细设计。所有设计中的考虑都应以设计说明书的形式加以详细描

述，以供后继工作使用并提交审查。

④ 编码。也就是编写程序。在这个阶段，要将软件设计转换成计算机可以接受的程序，即写成以某一程序设计语言表示的“源程序清单”。

⑤ 测试。在这个阶段，要通过测试去检查软件的各个组成部分的正确性，这也是保证软件质量的重要手段。首先要进行单元测试，以发现模块在功能和结构方面的问题，其次将已测试过的模块组装起来进行组装测试。最后按所规定的要求，逐项进行有效性测试，确定已开发的软件是否合格，能否交付给用户使用。

⑥ 维护。在软件投入正式使用后，便进入了维护阶段。软件在运行中可能由于多种原因，导致一些错误，需要对它及时进行修改。另外，由于外部环境的变化，也可能要对软件进行必要的更改。

2．原型法

• 原型法的特点

作为 MIS 的一种开发方法，原型法从原理到流程都很简单。它有如下特点：

① 它的开发过程是一个循环往复的反馈过程。开始，用户和设计者对于所设计系统的要求和功能的认识是不完整的、粗浅的。但在原型法的开发过程中，通过建立原型、演示原型及修改原型的循环过程，使得 MIS 逐步达到所期望的目标。

② 原型法在系统分析的初期阶段就引入模拟的手段。首先根据软件人员对用户要求的理解，模拟出一个系统原型，然后就这个模型开展讨论。这样用户很快就可以得到他们所要求的系统的模型，接触和使用模型系统，从而缩短了用户与软件技术人员的距离。

• 使用原型法的优点

① 所有问题的讨论，都围绕某一个确定的模型进行，彼此之间联系紧密。

② 通过对原型的接触和使用，能够启发开发人员去发掘问题，从而不断地修正、完善模型，最终得到一个理想的系统。

③ 原型法的开发周期短、使用灵活、容易修改，这对于管理体制不够稳定的系统更加适合。

• 原型法的开发阶段划分

① 确定系统基本要求。用户和系统分析人员共同进行调查、分析，得出用户对系统的基本要求，如数据规范、屏幕样式、输出形式等。但是这只要求对系统有一个基本的了解，不必对系统进行详细分析，也不必写出详细说明。这样得到的功能需求是不完全的、有缺陷的。这种不完全性在后续阶段将得以弥补。

② 构造初始原型。对所设计的系统有了初步了解以后，就要设计一个初步的模型——原型。原型要求能满足基本的要求，它是初步的系统。

③ 演示原型。有了初始的原型后，设计者就应该对用户演示这个原型，并在演示后征求用户对原型的评价及要求，这时面对用户的仍是未来系统的模型。

④ 修改原型。通过演示模型后，从用户那里得到反馈意见。假若用户对原型不尽满意，则必须对原型作出修改，然后再演示，直至用户满意为止。

⑤ 运行维护。对初步构造的模型，经过演示评价后，作出必要的修改，便进入运行阶段。在运行阶段，还可能会暴露出一些问题，仍需不断地更正。

1.1.2 管理信息系统开发的策略

MIS的开发是一项复杂的系统工程，需要花费大量的人力、物力及财力，而往往开发质量还不高，用户不满意，这通常是因为信息需求的不确定性造成。因此，在开发之前，如何根据信息需求中的不确定因素，选择一个合适的开发策略便显得十分关键。

在确定信息需求的过程中，之所以存在着不确定性，这首先是因为用户本身提出的要求未能做到完全、精确；其次开发人员也难以透彻理解开发任务的全部含义；最后的也是最主要的原因是信息需求本身是动态的，会随着外部环境的变更而发生变化。

1．影响信息需求不确定性的因素

影响信息需求不确定性的因素主要有以下几点：

- 项目大小。这是指项目功能的多少，以及项目数量的大小，项目所耗时间的多少，项目花费成本的大小。项目越小，不确定性程度就越低。
- 结构化的程度。这主要是指人工管理的结构化程序，具体包括决策过程的结构化程序、事务处理的规范化程序以及管理体制的确定性程序。结构化程序越高，不确定性的程序就越低。
- 对任务的理解程度。这又包括两个方面，首先是用户对开发任务本身能否理解得透彻，其次是指开发人员对任务的目的能否完全理解。理解得越透彻，不确定性程度就越低。

2．开发策略的选择

根据信息需求的不确定性因素，选取相应的开发策略。

① 接收式的开发策略。这种策略的含义是完全按照调研过程中确定的信息需求进行开发。只有当用户对信息需求的叙述完全准确、固定时，才可选取这种开发策略。

② 直线式的开发策略。这种开发策略的含义是：从需求定义到最后的开发，直线地进行下去。每完成一步后，采用一些步骤来核对，以保证与需求一致。当信息需求能较好地确定，在开发过程中很少或不修改的前提下，方可采用这种开发策略。

③ 迭代式的开发策略。当信息需求的确定性程度较低时，适宜采用这种策略。它的具体含义是：每当发现需求有错误或不适当时，要回到需求确定过程，与用户一起修改系统说明书。

④ 实验式的开发策略。这种策略的主要含义是：通过用户实际使用系统的经验来使需求的准确性得到保证。当信息需求的确定性很低时，宜采用这种开发策略。

⑤ 规则式的开发策略。规则式开发策略的含义是：在开发之前，要对系统作总体规划。这种开发策略是目前比较常用的。

1.1.3 精干的开发队伍

对于任何一个MIS，不管其规模大小如何，也不管相应的管理水平高低，它的实现都是靠MIS的开发人员。MIS开发人员的素质在很大程度上决定了MIS的质量。MIS的开发需要五类人员：

① 用户：在这里的用户，是指业务分析员，他们也是MIS的最终用户。MIS的信息需求，全部由用户提供。因此主用户也是MIS开发队伍中的组成成分。

② 系统分析员：系统分析员是MIS技术上的总设计师。MIS成败的关键往往就取决于系统分析员。作为MIS的系统分析员，至少应具备如下能力：

- 能协调MIS近期目标与长远目标之间的关系。
- 能调度各种技术手段。
- 能组织应用项目的实施。

③ 高级程序员：高级程序员是MIS的主要实施者，是MIS开发的骨干力量。在系统分析员的指导下，全面负责各子系统的设计、编码、测试和安装。

④ 程序员：程序员的主要任务是在高级程序员的指导下，进行详细设计、编码及测试工作。

⑤ 操作员：操作员的主要任务是为程序员及高级程序员录入源程序，还应该具备一定的调试能力。有时也兼做录入员的工作，录入数据及日常的文字处理等。

1.2 Visual FoxPro 程序设计的特点

在开发人员明确需求分析之后，接下来就可开始进行程序设计了。从概念上讲，程序设计就是为了完成某一具体任务而编写一系列指令；从深层上看，Visual FoxPro 程序设计涉及到对存储数据的操作。通常，在程序中能够完成的工作都可以通过人工操作来完成，但需要足够的时间。例如，要在 customer（顾客）表中要查找一个特定顾客的信息（如 Ernst Handel 公司），可以通过人工执行一系列指令来实现：

① 在“文件”菜单中选择“打开”命令。

② 在“文件类型”列表框中选择“表”。

③ 进入 Visual FoxPro 目录下的 samples \ data \ 子目录，在文件列表中双击 CUSTOMER.DBF。

④ 在“显示”菜单中选择“浏览”命令。

⑤ 滚动浏览表，查找 Company 字段为“Ernst Handel”的记录。

若要通过编程实现上述目的，可以在“命令”窗口中键入下列 Visual FoxPro 命令：

```
USE samples \ data \ Customer
LOCATE FOR Company = "Ernst Handel"
BROWSE
```

在表中找到这家公司以后，就可以执行预定的操作了。

若要通过人工操作提高订单的最高限额，可按如下步骤来进行：

① 选定 maxordamt 字段。

② 将 maxordamt 字段值乘以 0.03，再加上原来的字段值，然后在该字段处键入得到的结果。

若要通过编程实现以上结果，可以在“命令”窗口中键入以下命令：

```
REPLACE maxordamt WITH maxordamt * 1.03
```

如果要修改某个顾客订单的最高限额，用人工方式或在“命令”窗口中键入指令都很容易实现，但如果要把所有订单的最高限额都提高 3 个百分点，恐怕就不是一件轻而易举的事情。若还是用人工方式来做，不但费时费力，而且还容易出错。解决此类问题的更好办法是编写一个可执行的程序文件，那么该文件可以轻松无误地完成这一工作，如下所示：

```
* 增加所有顾客订单最大限额值的示例程序
USE samples \ data \ customer      && 打开 CUSTOMER 表
SCAN      && 浏览表中所有记录，针对每条记录执行 SCAN 与 ENDSCAN 之间的所有指令
    REPLACE maxordamt WITH maxordamt * 1.03      && 将订单的最高限额提高 3%
ENDSCAN      && 结束由 SCAN 开始、针对表中所有记录执行的指令序列
```

和“命令”窗口中单独键入每条命令相比，运行程序有如下优点：

- 使得程序具有更强的针对性：由于 Visual FoxPro 是一个通用的数据库管理系统，因此它更适合作为一开发平台。要让普通用户理解其用法，既十分困难也完全没有必要。
- 不可能让每个 MIS 用户都去购买一套 Visual FoxPro 系统，这从经济上和效率上都是不允许的。
- 此外，程序还具有修改方便的特点。假定程序的某些功能需要调整，或者需要增强，用户只需对程序稍加修改即可。

Visual FoxPro 的开发环境主要有两个作用：

- 辅助程序开发和调试，例如，当我们书写了一段程序后，希望看一看程序结果是否正确，此时即可借助 Visual FoxPro 的菜单和工具栏来查看。此外，请大家特别留意，Visual FoxPro 的菜单是动态变化的，即在各种不同的情况下，系统将打开不同的菜单。例如，当用户打开一个表时，系统将在主菜单中增加一“表”项。
- 执行某些简单的数据管理工作。例如，用户在打开一个表后，可用“表”菜单定位记录、追加记录等。

1.2.1 规划应用程序

Visual FoxPro 应用程序通常由以下几部分组成：一个或多个数据库、设置应用程序系统环境的主程序以及用户界面（诸如表单、工具栏和菜单等）。此外，还可以包括查询和报表，它们允许用户检索或输出自己的数据。

认真细致的规划可以节省时间、精力和资金。在规划阶段，应该让最终用户更多地参与进来。无论多么仔细的规划，在项目实施过程中也需要不断润色，并接受最终用户的反馈。

在开发之前所做的设计方案往往会对最终结果产生很大的影响。许多问题都应在深入开发之前加以考虑，例如：这个应用程序的用户是谁，用户的主要操作是什么，要处理的数据集合有多大，是否要使用后台数据服务器，以及是单用户还是网络上的多用户等等。

1. 一般的用户操作

您的最终用户可能需要处理顾客、订单或各种产品，他们处理信息的方式将决定应用程序如何进行数据操作。例如，像 \ VFP \ SAMPLES \ TASTRADE.APP 程序所使用的订单输入表单也许对一些应用程序有用，但这个表单可能并不适用于管理货物清单或追踪销售记录。

2. 数据库的大小

当需要处理庞大的数据集合时，考虑最多的恐怕是如何提高性能。有时，您可能需要调整用户在数据之间移动的方式。例如，如果一个表中只有二三十个记录，那么记录指针一次只移动一个记录不会有什么问题，但如果表中有二、三万个记录，就必须为用户提供能找到

所有数据的其他方法（比如增加搜索列表、对话框、筛选和定制查询等）。

3．单用户和多用户

创建应用程序时，最好考虑到几个用户同时访问数据库的情况。Visual FoxPro 提供了一些技术，使您能够很容易地进行共享访问方面的编程。

4．国际化考虑

若事先知道应用程序仅使用在单一语言环境中，可以不考虑国际化问题。另一方面，如果您想扩大市场，或用户要处理国际化的数据和环境设置，则应在创建应用程序时考虑这些因素。

5．本地数据和远程数据

若应用程序需要处理远程数据，则必须使用新的管理方法，这种方法与处理本地 Visual FoxPro 数据有所不同。

对于以上各种问题，我们将在以后各章分别进行介绍。

1.2.2 程序开发过程

在创建应用程序的过程中，有许多重复性工作。但由于两个应用程序不可能完全相同，因此您需要在开发出原型的基础上，对其各组成部分不断优化，以得到一个完善的产品。最终用户的要求和期望有可能改变，那么应用程序的某些方面也应随之改变。所有人在编程时都会犯些小错误，所以在测试和调试过程中，可能会需要重新设计和改写某些代码。创建应用程序的过程大致如图 1-1 所示。

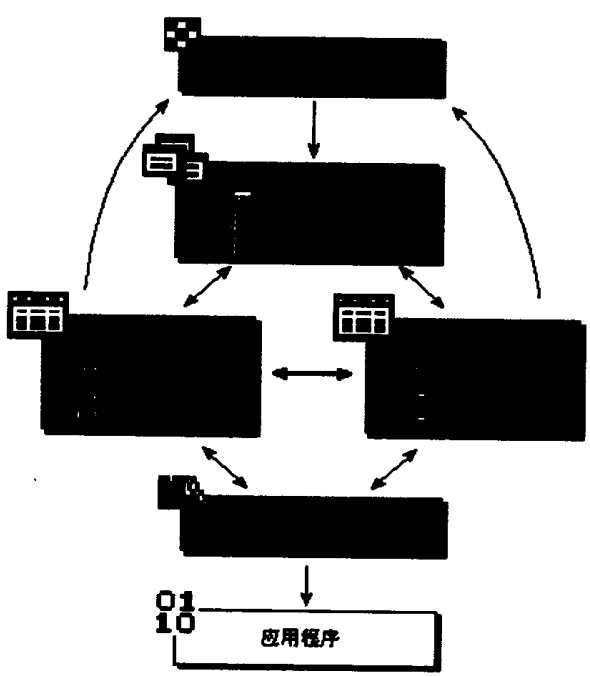


图 1-1　程序开发过程

除了考虑以上的整体过程外，还需要仔细推敲应用程序中应包含哪些功能，涉及到哪些数据以及如何构造数据库的结构等问题。您需要设计一个用户界面，使用户可以访问应用程序。同时，创建一些报表和查询，以便用户可以从数据中提取有用信息。

1. 开始开发工作

在计划好应用程序中所需组件后，可能会希望建立一个目录框架和项目以组织那些为应用程序而建立的组件文件。可在“Windows 资源管理器”中建立框架，在“项目管理器”中创建项目，或使用“应用程序向导”同时建立。在“向导”完成工作之后，您还可以用“项目管理器”或其他 Visual FoxPro 设计器进一步组织项目和组件。

2. 使用“项目管理器”

“项目管理器”能够编译已完成的应用程序。此外，在开发阶段，它还可以使应用程序某些组件的设计、修改和运行变得容易。“项目管理器”提供了以下功能：

- 双击应用程序组件（表单、菜单、程序）以运行或进行修改。
- 类、表或字段可被直接从“项目管理器”拖入“表单设计器”或“类设计器”。
- 可在类库之间拖动类。
- 可以方便地查看和修改自己的表。
- 可为自己的应用程序组件添加说明。
- 可在项目之间拖放各种条目。

有关使用“项目管理器”的详细内容，请参阅与本书配套的《中文 Visual FoxPro 5.0 基础教程》一书。

3. 创建数据库

因为数据库应用程序在很大程度上依赖于所管理的数据，所以最好从数据入手进行应用程序的设计。在动手设计用户界面和用于管理数据的组件之前，请设置数据库，并确定表之间的关系以及所希望的事务规则等信息。在可靠的数据库基础上，开发工作将会变得容易许多。

4. 创建类

您有时不需要自己创建类，只用 Visual FoxPro 的基类就可以创建一个可靠的面向对象的事件驱动程序。偶尔您也许想自己创建一些类，实现一些特殊功能。Visual FoxPro 可靠的类库能帮助您快速创建原型，并向应用程序中添加功能，使代码更易管理和维护。使用“表单设计器”（使用“文件”菜单中的“另存类”命令）或“类设计器”，您可以在程序文件中创建类。

5. 设计用户操作界面

界面直接表现一个应用程序的功能。用户对应用程序是否满意，很大程度上取决于界面是否友好。也许您的类模型很简洁，代码很精致，解决难题的方法很巧妙，但这一切用户都看不到，他们所能见到的只是您提供的用户界面。Visual FoxPro 的设计工具使得创建富有吸引力并且功能丰富的界面成为一件轻松愉快的事情。

用户界面主要包括表单、工具栏和菜单，它们可以将应用程序的所有功能与界面中的控件或菜单命令联系起来。

6. 设计访问信息的方法

也许您想在表单上为用户显示一些信息，甚至想给用户提供一些方便，准确给出他们所需的内容，让他们自己选择是否把信息打印到报表或标签上。查询（特别是能够接受用户自定义参数的查询）可以扩展用户控制数据的能力，报表则允许用户选择如何打印数据结果（可以全部打印、部分打印或概要打印），ActiveX 控件和 OLE 自动化允许应用程序之间共

享信息和功能。

7. 测试和调试

测试和调试是开发人员在开发工作的每一步中都需要做的事。随着工作的深入，您最好不断进行测试和调试。假定您创建了一个表单，那么在处理应用程序的其他部分之前，最好检查一下表单能否完成预定的功能。

1.2.3 程序设计示例

掌握了一些基本概念后，程序设计就变成了一个不断重复的过程。您需要多次重复某些步骤，并在这个过程中不断对代码进行优化。从编写第一行程序开始，便不断进行测试，完成每个“试验—查错”的过程。对语言越熟悉，则编程速度越快，而且更多的初步测试工作将在头脑中进行。

程序设计的基本步骤包括：

- 对问题进行说明。
- 分解问题。
- 编制各模块。
- 测试并完善各模块。
- 组装全部模块。
- 整体测试。

开始程序设计之前，请注意以下几个问题：

- 在解决问题之前，必须把问题说明清楚，否则会不断进行修改，丢弃已编好的代码并从头再来，而且最终也不可能得到满意的结果。
- 将问题分解成可单独处理的几个步骤，而不是一下子解决全部问题。
- 在开发过程中不断测试和调试已编好的代码。通过测试检查代码是否能实现所需的功能；调试是找出代码在哪里出错并纠正这些错误。
- 精炼数据和数据存储方式，便于程序对其进行处理。这需要正确构造表格的结构。

本节以下部分将详细介绍建立一个小型 Visual FoxPro 程序的各个步骤。

1. 对问题进行说明

开始解决问题以前，需要将其说明清楚。有时调整问题的说明方式会有助于问题的解决。

假设从不同的数据源获得一批数据，其中大部分是数值型数据，但有些数据中，除包含数字字符外，还夹杂着一些虚线和空格。现在需要从字段中清除这些空格和虚线，并将所得到的数值型数据加以保存。

可以按这种方式说明需要这个程序达到的目标，而不是简单地立即着手从原始数据中除去空格和虚线：用新值替换字段中的原始值，新值包含原始值中除空格和虚线外的所有内容。这种说明方式的优点在于可以避免在处理可变长度字符串时遇到的困难。

2. 分解问题

因为最终是以操作、命令和函数的方式将具体的指令提供给 Visual FoxPro，所以需要将问题分解为多个独立的步骤。在本问题中，最容易分离出来的任务是在字符串中逐个扫描字符，只有将单个字符分离出来才能决定是否应该保存它。

在扫描字符的时候，将检查它是否为虚线或空格。到了这一步也许您想要更详细地说明这个问题，括号之后的数据应该如何提取，货币符号、逗号和句号该如何消除等一系列问题将出现在脑海中。代码越通用，为将来节省的工作量就越多，问题就在于如何做到事半功倍。下面给出了一个比从前适用范围更广的问题说明方式：用原始数据中的数值型字符更新字段值。

这便是在字符的层次上重新说明问题。如果字符是数字则保留，否则将指针移向下一个字符。当用原始字符串中的数字重新构成一个只含有数字元素的新串时，将这个新串替换掉原来的字符串。如此循环，直到将所有记录中的数据全部更新。

概括起来，全部问题可以分解成以下一些模块：

- 逐个检查字符。
- 判断该字符是否为数字。
- 若是数字，则将其复制到新串。
- 检查完字符串中的全部字符后，用只含数字的新串替换原串。
- 重复上述步骤，直到表中全部记录都被更新。

3. 编制模块

清楚了要达到的目标以后，便可以开始使用 Visual FoxPro 的命令、函数和操作符来构造各模块。如果已明确了所需的命令或函数，便可以查看“帮助”，了解这些命令或函数的语法格式。如果仅仅知道需要做什么，而不清楚应该使用什么命令或函数，请搜索“帮助”中的“语言分类”。

因为命令和函数是用来处理数据的，所以需要一些数据来测试其功能。这些用于测试的数据应与实际数据尽量相近。

例如，在“命令”窗口中键入如下命令，将一个测试字符串赋值给变量：

```
cTest = "123-456-7 89 0"
```

- 逐个检查字符

首先，需要在字符串中检查单个字符。请参阅“字符函数”，从中查看用于操作字符串的函数列表。您将发现三个用以返回字符串特定部分的函数：LEFT ( )、RIGHT ( ) 和 SUBSTR ( )，其中 SUBSTR ( ) 可以返回字符串中的任意子串。

SUBSTR ( ) 需要三个参数：字符串、起始字符的位置和需返回的字符数。若要了解 SUBSTR ( ) 是否可完成所需工作，请在“命令”窗口中键入以下命令：

```
? SUBSTR (cTest, 1, 1)
? SUBSTR (cTest, 3, 1)
? SUBSTR (cTest, 8, 1)
```

其输出应该为：

```
1
3
-
```

用于测试的字符串中的第一个、第三个和第八个字符将显示在 Visual FoxPro 的主窗口中。

可以使用循环结构多次重复执行相同操作。因为用于测试的字符串有固定的字符数

(14)，所以能够使用 FOR 循环语句。每执行一遍，FOR 循环的计数器增 1，因此可在 SUBSTR ( ) 函数中使用该计数器。因为在“命令”窗口中不能测试循环结构，所以我们编写一个示例程序，其步骤如下：

- 在“命令”窗口中键入如下命令：

```
MODIFY COMMAND numonly
```

- 在打开的窗口上键入如下代码：

```
FOR nCnt = 1 TO 14
    ? SUBSTR (cTest, nCnt, 1)
ENDFOR
```

创建程序之后便可运行。

运行程序的步骤如下：

- 在打开的程序窗口中，按下 CTRL+E 键。
- 若“保存”对话框出现，选择“是”按钮。

运行此程序时，各个结果将在 Visual FoxPro 主窗口中分行显示。

- 判断字符是否为数字

从字符串中分离出单个字符之后便可判断它是否为数字。在“字符函数”帮助主题中，ISDIGIT ( ) 可以完成此项功能。

在“命令”窗口中，键入以下命令来测试 ISDIGIT ( )：

```
? ISDIGIT ('2')
? ISDIGIT ('-')
? ISDIGIT (SUBSTR (cTest, 3, 1))
```

其输出如下：

```
.T.
.F.
.T.
```

由输出结果可以看出：在 cTest 中，'2'是数字，'-'不是，'3'是数字。

- 若字符为数字，将它复制到新串中

既然可以逐一检查字符是否为数字，下面我们可以使用变量 cNumOnly 来保存该数字。为了创建一个变量，需要给它赋一个初值：一个零长度的字符串。

```
cNumOnly = ""
```

因为 FOR 循环扫描整个字符串，所以最好用一个变量来暂存每次分离出来的字符以便处理。

```
cCharacter = SUBSTR (cTest, nCnt, 1)
```

注意 *在计算、求值以及函数中得到的结果最好保存到内存变量中，这样便于直接处理这些变量而不必重新计算或求值。*

可以使用以下代码将每个被确认为数字的字符添加到新串中：

```
cNumOnly = cNumOnly + cCharacter
```

这个程序是：

```
cNumOnly = ""
FOR nCnt = 1 TO 14
```

```
        cCharacter = SUBSTR (cTest, nCnt, 1)
        IF ISDIGIT (cCharacter)
            cNumOnly = cNumOnly + cCharacter
        ENDIF
    ENDFOR
```

4. 测试模块

如果将两行用于输出字符串的命令加在程序末尾并运行程序，就可以看出程序如何对被测试的字符串进行操作。

```
cNumOnly = ""
FOR nCnt = 1 TO 14
    cCharacter = SUBSTR (cTest, nCnt, 1)
    IF ISDIGIT (cCharacter)
        cNumOnly = cNumOnly + cCharacter
    ENDIF
ENDFOR
? cTest
? cNumOnly
```

其输出如下：

```
123-456-7 89 0
1234567890
```

输出似乎是正确的，但在测试模块的过程中改变被测试的字符串将出现问题。在“命令”窗口中，键入以下命令再运行程序：

```
cTest ="456-789 22"
```

程序产生错误信息，FOR 循环应执行 14 次而测试串中只有 10 个字符。这样看来，对于可变长字符串需要用另外的方法。若再次查看帮助中“字符函数”将发现，函数 LEN( ) 将返回字符串中字符的个数。如果把它放到 FOR 循环语句中，那么程序在两种情况下都可以良好地运行。

```
cNumOnly = ""
FOR nCnt = 1 TO LEN (cTest)
    cCharacter = SUBSTR (cTest, nCnt, 1)
    IF ISDIGIT (cCharacter)
        cNumOnly = cNumOnly + cCharacter
    ENDIF
ENDFOR
? cTest
? cNumOnly
```

5. 组装全部模块

若要完善此程序，则需要从表中读取数据。在对表操作时，需要逐条扫描记录，针对记录中的字段而非变量进行操作。

可以先建立一个临时的表保存一些示例字符串，此表可以只含有一个字符型字段 Test-Field 和四五个记录。